양방향 송수신

서버-클라이언트 프로그램

개발 보고서

**SW개발2팀**

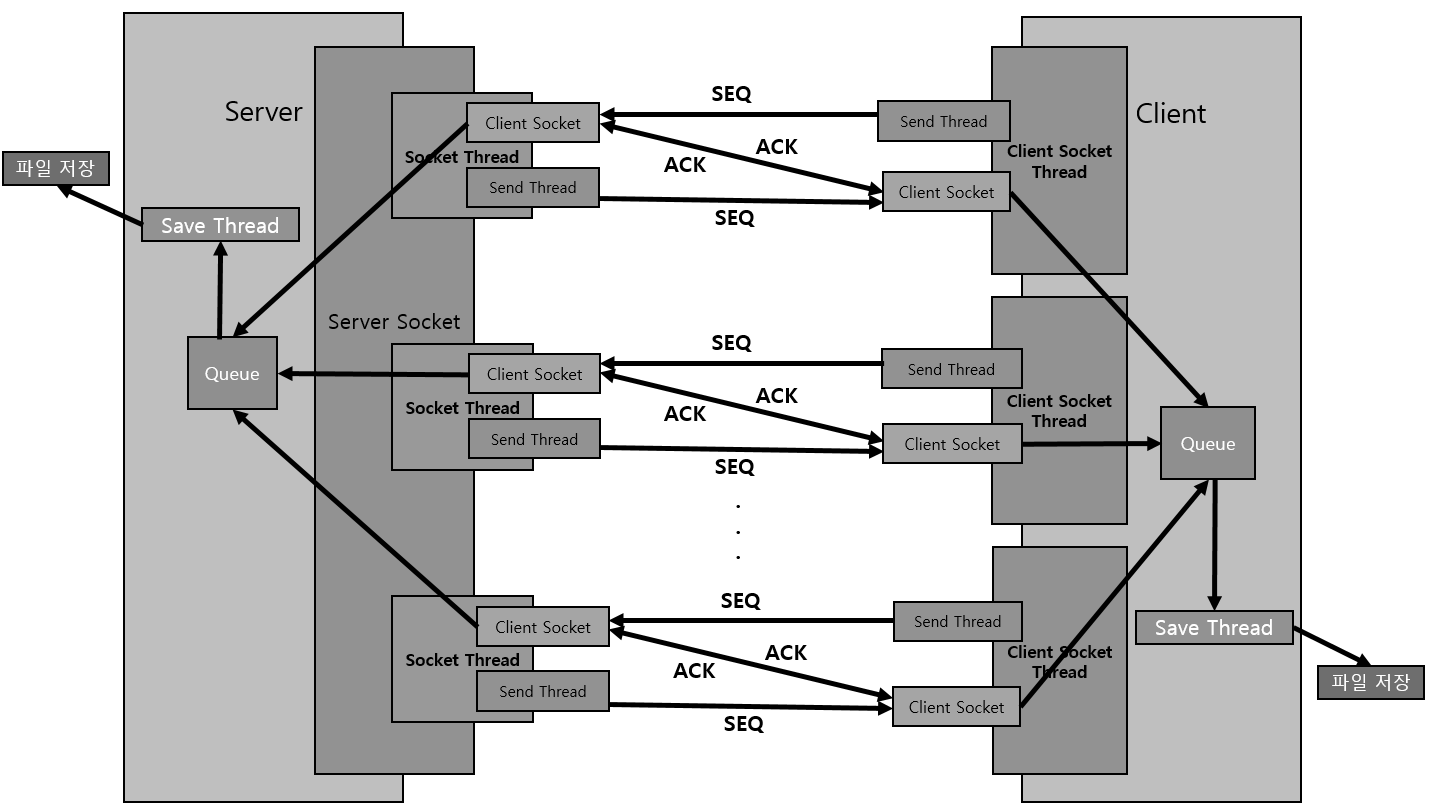
**이민우 사원**

**목차**

1. **프로그램 개요**
2. **화면 구성**
   1. 메인 화면
   2. Control Panel
3. **파라미터 설정**
   1. 전송 스레드 파라미터 설정
   2. 파일 저장 파라미터 설정
4. **서버-클라이언트 연결 및 프로그램 동작**
   1. 클라이언트 생성
   2. 메시지 규칙
   3. 오류 로그
   4. 파일 저장
5. **테스트 결과**

**1. 프로그램 개요**

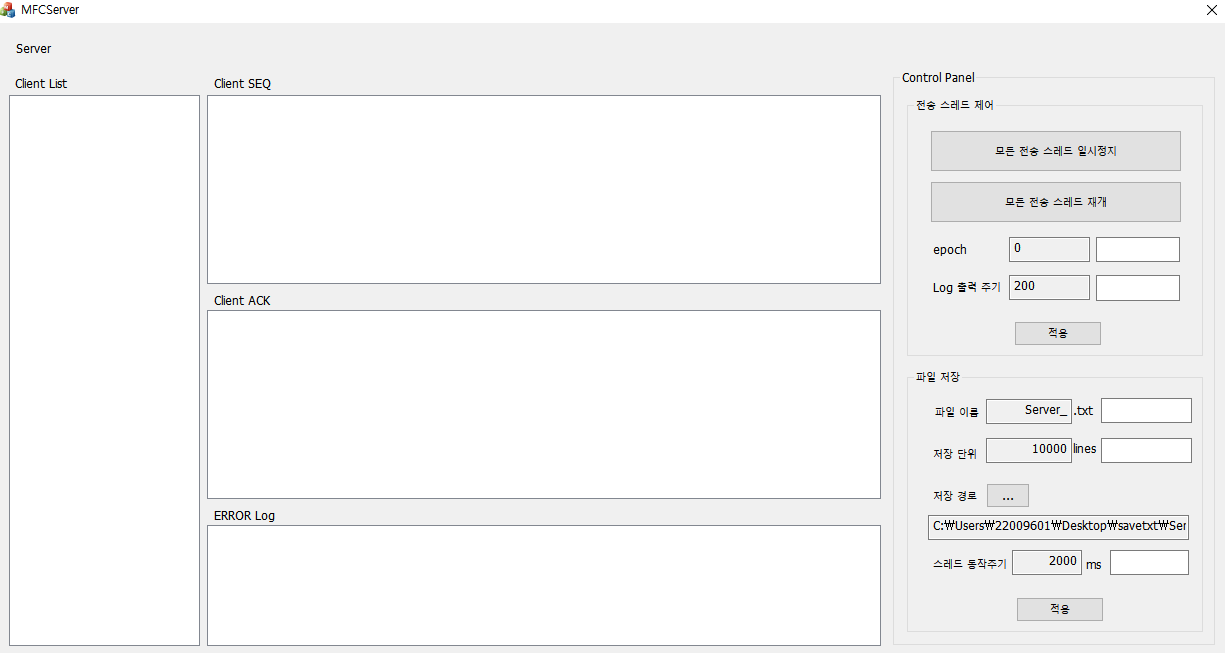
서버-클라이언트 소켓 프로그램으로서 C++ MFC로 일대다, 양방향 송수신을 구현한 프로그램이다. 서버와 클라이언트는 Localhost(127.0.0.1), 21000번 포트를 통해 통신한다. 클라이언트 프로그램은 하나의 프로그램에서 클라이언트 객체를 여러 개 생성함으로써 다수의 클라이언트 생성을 구현하였다. 서버, 클라이언트는 독립적인 송신 스레드를 가지며 15ms마다 2048bytes 크기의 송신 데이터(SEQ)를 송신하고 그와 동시에 수신한 송신 데이터(SEQ)에 대한 응답(ACK)을 전달한다. 서버와 클라이언트는 정상적으로 수신한 메시지를 큐(queue)에 저장하였다가 큐의 길이가 사용자가 설정한 저장 단위보다 커질 때 큐에서 저장 단위만큼 메시지를 꺼내어 텍스트 파일로 저장한다. 비정상 수신일 경우 발생하는 오류 로그는 별도의 텍스트 파일에 저장한다. 다음은 프로그램의 구조이다.

****

**2. 화면 구성**

**2.1 메인 화면**

서버, 클라이언트 프로그램을 실행하면 다음과 같은 메인 화면이 생성된다. (클라이언트는 클라이언트 생성 버튼이 추가로 있다.)



**5**

**4**

**3**

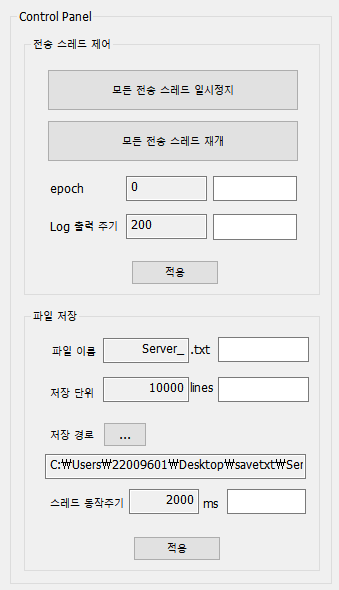
**2**

**1**

1. Client List : 연결된 클라이언트의 리스트를 표시한다.
2. Client SEQ : 연결된 클라이언트로부터 받은 SEQ 로그 데이터를 표시한다.
3. Client ACK : 연결된 클라이언트로부터 받은 응답(ACK) 로그 데이터를 표시한다.
4. Error Log : 수신 메세지에서 발생하는 모든 오류 로그를 표시한다.
5. Control Panel : 파라미터 설정 및 스레드를 제어하는 패널이다.

**2.2 Control Panel**

Control Panel은 다음과 같이 구성하였다.



**10**

**9**

**8**

**5**

**6**

**7**

**4**

**3**

**2**

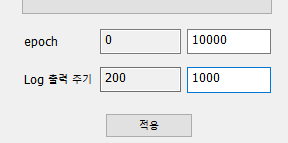
**1**

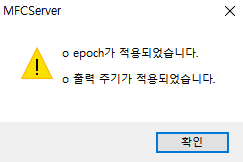
1. 모든 전송 스레드 일시정지 버튼 : 서버의 모든 전송 스레드를 일시정지한다.
2. 모든 전송 스레드 재개 버튼 : 서버의 모든 전송 스레드를 재개한다.
3. Epoch : 전송 스레드가 몇 번 전송하는지에 대한 값을 입력한다. Epoch만큼의 데이터를 전송하고 전송을 중지하며 스레드 재개 버튼을 누르면 다시 epoch만큼 전송하고 중지한다. 기본값은 0이고 값이 0이면 무한히 전송한다.
4. Log 출력 주기 : 로그 출력 화면에 몇 번의 주기로 로그를 출력할 지에 대한 값을 입력한다. 기본값은 200이다.
5. 적용 버튼 : 전송 스레드 제어에 대한 모든 값을 설정하는 버튼이다.
6. 파일 이름 : 저장할 파일이름을 입력한다. 기본값은 Server\_, Client\_ 이다.
7. 저장 단위 : 1개의 파일에 저장될 데이터의 양에 대한 값을 입력한다. 2000 ~ 100000사이의 값을 입력할 수 있고, 기본값은 10000이다.
8. 저장 경로 : 저장 경로를 설정하는 버튼이다. 클릭 시 파일 다이얼로그가 생성된다.
9. 스레드 주기 : 저장을 담당하는 스레드가 동작하는 주기를 입력한다. 최소값은 1000이며, 기본값은 2000 이다.
10. 적용 버튼 : 파일 저장에 대한 모든 값을 설정하는 버튼이다.

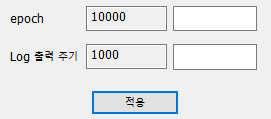
**3. 파라미터 설정**

**3.1 전송 스레드 파라미터 설정**

Epoch와 Log 출력 주기를 입력하고 적용 버튼을 누르면 각 파라미터 값이 적용된다. Epoch가 0일 경우, 무한히 전송한다.

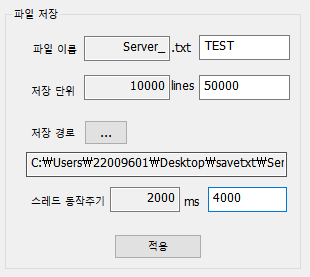




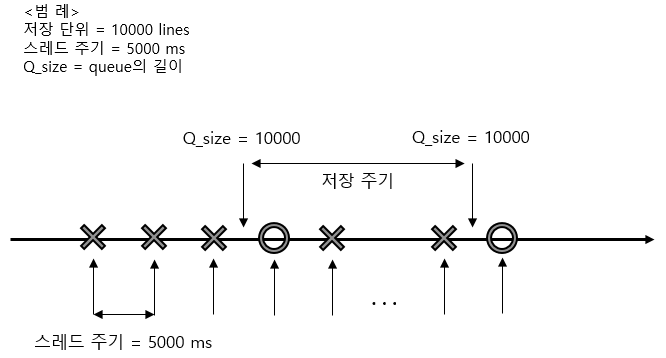


**3.2 파일 저장 파라미터 설정**

파일 이름, 저장 단위, 스레드 동작 주기를 입력하고 적용 버튼을 눌러 각 파라미터 값을 적용한다.

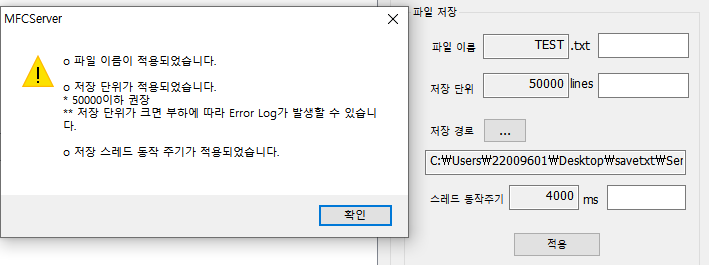


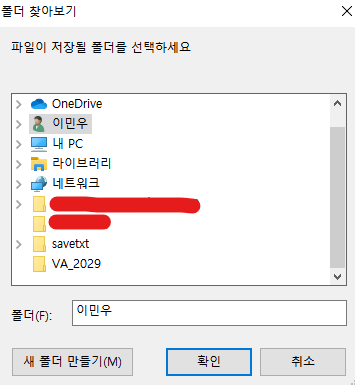
저장 단위는 1개의 text파일에 저장될 라인 수를 설정한다. 저장 단위가 클수록 저장 스레드가 발생시키는 CPU 부하가 커지기 때문에 상황에 따라 오류 로그가 발생할 수 있다. 따라서 2000~100000 lines단위로 범위를 제한하였다. 스레드 주기는 저장 스레드가 깨어나 동작하는 주기이다. 스레드 주기 값이 작으면 그 만큼 자주 깨어나 저장 조건을 검사하지만 부하가 더 발생될 수 있고, 값이 크면 저장이 지연될 수 있다. 저장 스레드는 설정한 스레드 주기마다 깨어나 X시점에서는 다시 휴면상태로 진입하고 O시점에서 저장 절차를 수행한다. 주의해야 할 점은 스레드 주기가 저장주기보다 커진다면 q에 데이터가 점점 쌓여 결과적으로 메모리 오버가 생길 수 있으므로 스레드 주기는 저장 주기보다 작게 설정해야한다.



**Time (ms)**

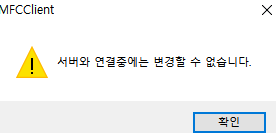
각 파라미터는 필수 입력사항이 아니며 입력이 없을 시 기본값으로 설정된다.





저장 경로 설정은 [ **…** ]버튼을 클릭하여 파일 다이얼로그에서 경로를 선택한다. 새 폴더를 만들어 경로를 만들 수 있고 원하는 경로를 클릭한 뒤 확인버튼을 눌러 설정을 마친다.

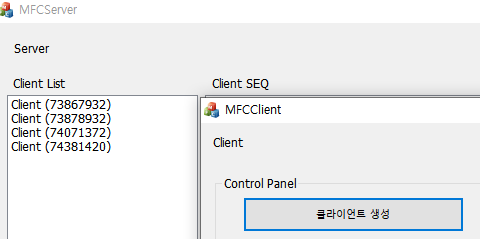
로그 출력 주기를 제외한 모든 파라미터는 서버-클라이언트 연결 중에는 변경할 수 없으므로 서버와 클라이언트의 파라미터 값을 모두 설정한 뒤 서버와 클라이언트를 연결한다.



**4. 서버-클라이언트 연결 및 동작**

**4.1 클라이언트 생성**

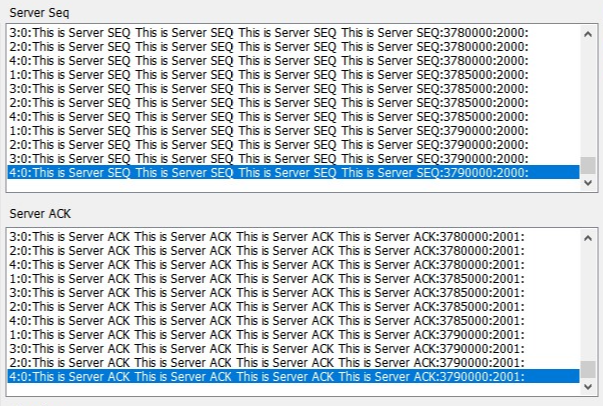
클라이언트 프로그램에서 생성을 원하는 클라이언트 수만큼 클라이언트 생성 버튼을 클릭한다.



클라이언트가 생성되면 서버와 클라이언트의 클라이언트 리스트에 연결된 클라이언트 리스트가 표시되고 클라이언트는 자동으로 서버에 연결 요청을 한다. 서버로부터 연결이 수락되면 서버와 클라이언트는 할당된 소켓을 통해 양방향 송수신을 시작한다.

**4.2 메시지 규칙**

다음은 클라이언트의 수신 로그이다.



송수신 메시지의 규칙은 ‘:’ 구분자를 이용하여 각 필드를 구분한다.

메시지 규칙은 다음과 같다.













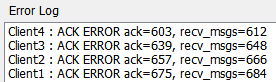
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 클라이언트 번호 | Undefined Flag | Text 데이터 | SEQ 또는 ACK번호 | 식별번호 |

**<식별번호>**

|  |  |  |
| --- | --- | --- |
|  | SEQ | ACK |
| 서버 | **2000** | **2001** |
| 클라이언트 | **1000** | **1001** |

**4.3 오류 로그**

오류 로그는 수신된 데이터에 대해 오류가 생긴 데이터의 로그를 보여준다.



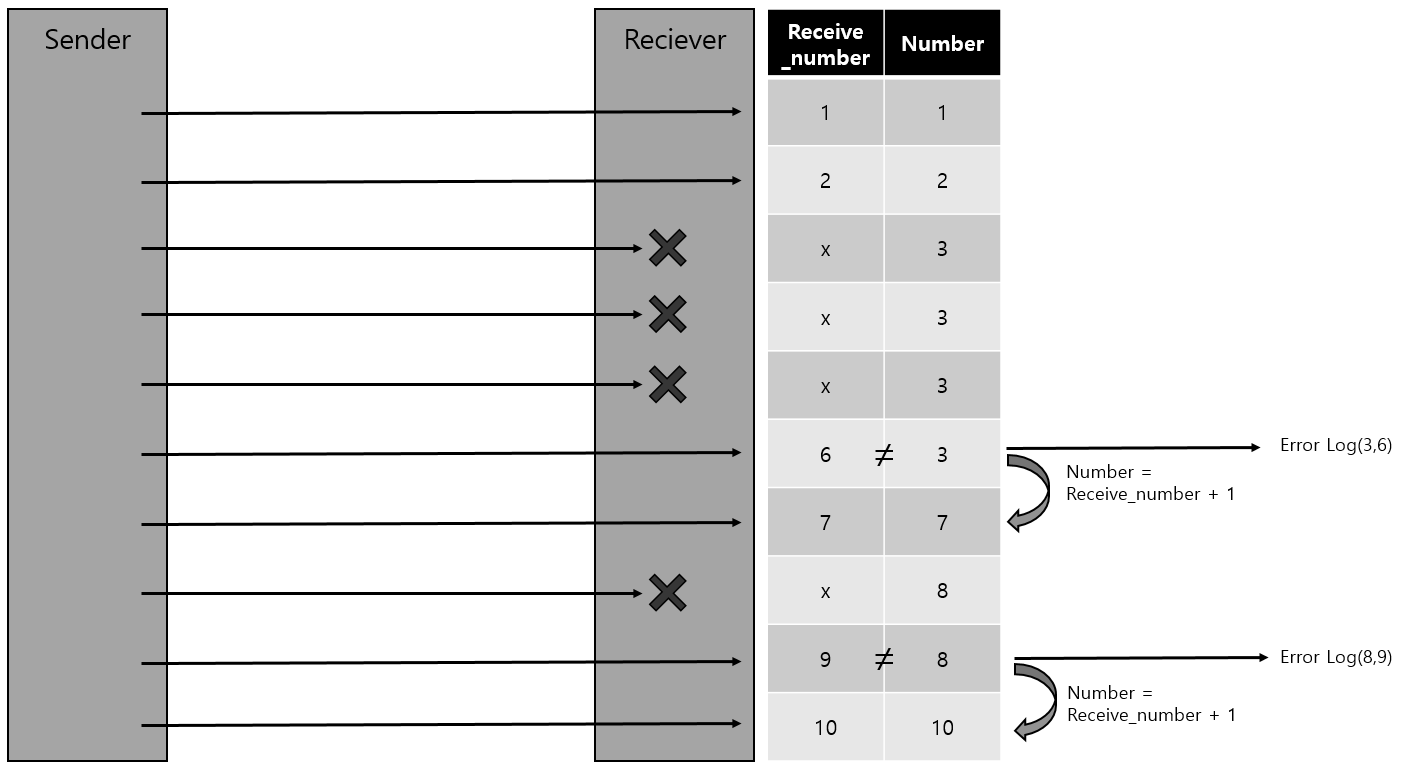






오류 로그에서는 오류가 발생한 클라이언트의 번호 정보가 맨 앞에 표시되고, ACK오류는 ACK ERROR, SEQ오류는 SEQ ERROR로 표시된다. 오류의 범위는 ack=정수A 또는 seq=정수A, recv\_msgs=정수B 형태로 표시되며 정수A ~ (정수B - 1)까지 이다. 예를 들어, 위의 오류 로그의 경우 클라이언트 4번에서 발생한 ACK 오류이며 ACK번호 603번부터 611번까지 문제가 생겼다는 것을 알 수 있다.

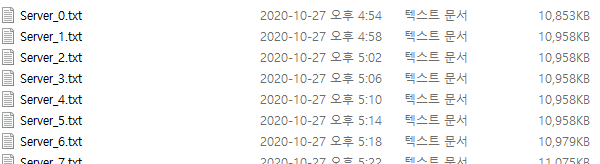
오류는 다음과 같은 방법으로 감지한다.

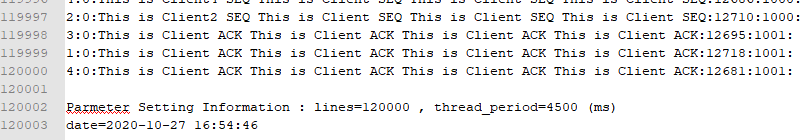


Receive\_number은 Sender로부터 받은 SEQ 또는 ACK번호이며 Number은 Reciever내에서 카운트되는 번호이다. 정상적인 수신일 경우 Receive\_num과 Number은 값이 같으며 Number은 1씩 증가하여 다음 번호를 기다린다. 비정상 수신일 경우 Receive\_number은 받은 내용이 없어 number와의 비교가 불가능해지고, Number은 카운트되지 않고 그대로 기존 값을 유지한다. Reveive\_num에 값이 정상적으로 수신되면 두 값을 비교하여 같지 않을 경우 오류 로그를 출력한다. 이때 Number은 Receive\_number + 1로 만들어 Number값 보정을 수행한다. 만약 보정을 하지 않는다면 이후에 오는 메시지는 모두 오류로 간주되기 때문에 Number값 보정이 필요하다.

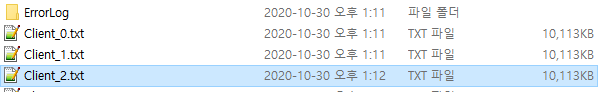
**4.4 파일 저장**

파일은 설정한 스레드 주기에 따라 저장 조건을 검사하며 저장 조건이 만족되면 정상적으로 수신한 데이터와 설정한 파라미터 정보, 시간 정보와 함께 설정 경로에 .txt 포맷 파일로 저장한다.



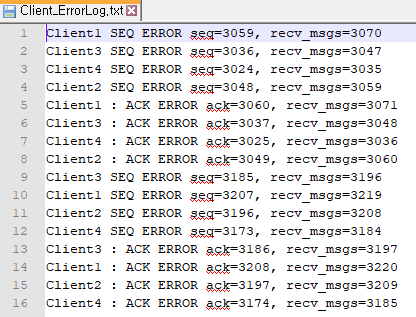


오류가 생긴 데이터에 대해서는 설정한 경로에 ErrorLog 폴더가 생성되며 그 안에 .txt 포맷 파일로 오류 로그가 저장된다.



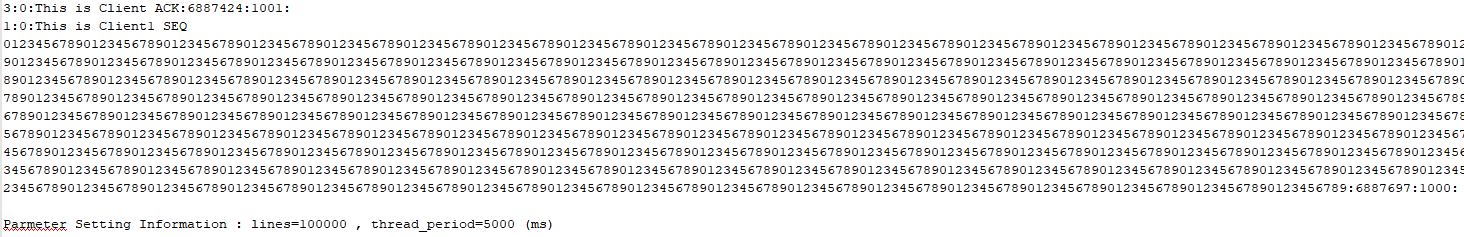




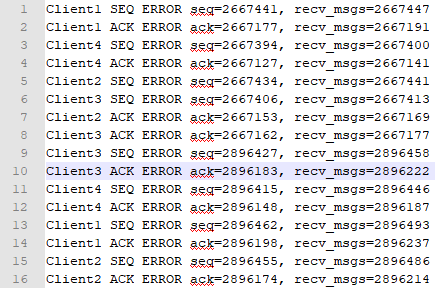


**5. 테스트 결과**

테스트는 4개의 클라이언트를 생성하여 2000자 이상의 길이를 가진 텍스트 메시지를 15ms 간격으로 송신하였고 약 30시간 정상 동작하였다.

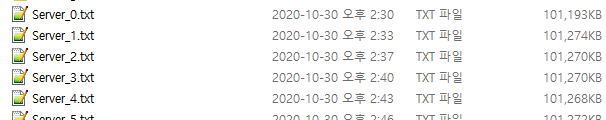


서버 측은 총 13,775,394개의 수신 메시지 중 366개의 오류 로그가 발생되었으며 오류 발생률은 약 0.002%를 보였다. 클라이언트 측은 284개의 오류가 발생하였고 오류 발생률은 서버와 비슷한 수준을 보였다. 저장 시퀀스가 수행될 때 저장 단위(lines)가 크면 클수록 파일 쓰기 과정에서 순간적인 CPU의 점유율이 높아지는데 이때 낮은 빈도로 오류 로그가 발생하는 것으로 파악된다. 따라서 하드웨어 사양이 좋아지거나, 저장 단위(lines)를 낮춘다면 오류는 감소할 것으로 기대된다.



336개

송신 메시지 길이 2000bytes 이상, 저장 단위는 10만 lines, 저장 주기는 5000ms일 때, 약 30시간 동안 저장된 로그는 서버, 클라이언트 각각 54.5GB로 총 109GB의 저장 공간이 사용되었다.



**. . .**

